

Initialization:

ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver & FPGA	PEU and DSP
100			Loaded and init itself then Create Instance of VSR (DAL)					Loaded at the time of system up	
			Init VSR by invoking Init() method that VSR Interface provides	VSR loaded					
102			VSR start broadcasting “KeepAlive”, Heart Beat with 1 second interval, packet to all PEUs using Control packet queue.				DD passes packet to FPGA		
								Upon receipt of “Keep Alive” packet, sends “Hello” packet to VSR indicating ID and other information such as number of ports, SN. For all PEUs	

ପାଦ କାହାର ମଧ୍ୟ ଥିଲା ?

ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver & FPGA	PEU and DSP
								DD passes packet to Shared memory	
			VSR creates PEU Interface Instance for each Physical PEU detected. Based on reported information, PEU Interface Instance provides table for ports.						
			Created						
ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver and DSP & FPGA	PEU and DSP
		PBX enumerates all PEUS and gets information of PEU through PEU Interface Instances those created to drive Physical PEUs. As PBX finds a matched Actual Port from the configuration, PBX invokes PEU Interface method to create and Actual Port Object for each specified port number.							
		PEU Interface Instance looks up specified port number in table and invokes Create Port Interface Instance as needed to create an Actual Port Object for each actual port specified by the PBX. Return handle of created Port Interface Instance (Actual Port Object) to PBX.							

ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver & FPGA	PEU and DSP
			VSR looks up WAV channel ID in its table and, if no WAV Port already created with that device ID, creates WAV Port Interface Instance (WAV port) and sets Stream ID to Device ID						
			VSR returns handle of created WAV Port Interface Instance						
			PBX provides Device ID to IVR application through TAPI						
	IVR App keeps Device ID for future use. It repeats for the number of WAV ports in its configuration								

004740 = 892820

Run Time:

ID	IWR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver & FPGA	PEU and DSP
104									CO Line Ring on one specific port
106									MC Detects Status, generates packet with PEU and Port info then sends packet to device driver
108								Device Driver sees packet in FPGA queue and copies to DAL by writing into shared memory	
110			VSR reads packets from the shared memory either in response to an event that generates a software interrupt or by polling device driver and receiving device pointers						
112			VSR dispatches message to the Actual Port Object that was created for port that detected CO Ring.						
			Actual Port Object passes the RING message to the PBX via COM Interface						

ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver &	PEU and DSP
----	---------	-----	-----------	------------	----------	-------------	------------	-----------------	-------------

								FPGA
130								FPGA in PEU scans PEU address then selects broadcast packets or packets addressed to the PEU and puts them in FIFO
132								MC sweeps out received packets from FIFO in PEU in response to interrupt or by polling.
134								MC writes control bit to appropriate SLIC for port with ringing CO line to cause it to "Go Offhook".
136								A Loop Current detector in the line interface circuit detects loop current. And the MC detects this change in status.
138								Loop Current status is reported to PBX by MC by upstream control packet through Device Driver and DAL.
								Pass it to DAL
								Dispatch message to addressed Actual Port Object
								Actual Port Object sends status change message to PBX

ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver & FPGA	PEU and DSP
140		PBX reports Port Status to IVR App through TAPI.							
144	IVR App invoke TAPI function requesting PBX to switch WAV Port of specified Device ID to couple it to the Actual Port Object mapped to the port that just went to "Offhook"	PBX invokes function of WAV Port Interface instance specified by the Device ID to connect to the actual port to listen.							
		Wav port with device ID specified by IVR app registers as listener to specified actual port.		WAV Port registered.					

DRAFT - 7/10 - 2023 - SIGN OFF

ID	IVR APP	PBX	VSR (DAL)	PEU WAV Port	Actual Port	WAV Driver	Device Driver & FPGA	PEU and DSP
146		PBX invokes function of Actual Port Interface instance to connect to WAV Port to listen.			Actual Port Object addressed by PBX registers as listener to specified WAV Port			

ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver & FPGA	PEU and DSP
									PEU start monitoring “play data” packet identified by Stream ID for the Port
148	PBX invokes TSP/TAPI function call to inform IVR App that WAV and Actual Port Objects have been coupled together								Any play data packets found with this stream ID are sent to Port DSP to mix with other signals such as TDM data and tone data.
150	IVR App invokes system function call to open prerecorded announcement file								
154	IVR App generates “WODM_OPEN” message with								

160	IVR App generates “WODM_PREPARE” message to WAV driver through OS with Device ID and size to provide buffer space in WAV Driver.						
162	IVR App generates “WODM_PREPARE” message to WAV driver through OS with Device ID and size to provide buffer space in WAV Driver – second buffer.						
164	IVR App. invokes system function call to read data from the opened file into the buffer. Then App generates “WODM_WRITE” message with Device ID and buffer address..						
	App repeats if buffer is available.						
ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver
							WAV Driver passes buffer to WAV Port designated by Device ID.
166					WAV Port 208 byte play data packet from the buffer		and sets PEU and Port

			address to all, (broadcast) and Stream ID. Write play data packet to Play Data Queue in shared memory. If given Play Data buffer is not empty, look for Play Data Queue then write the next packet upon queue availability.
168			FPGA generates interrupt every 26 msec for Device Driver to get one packet from the Play Data Queue in shared memory for each WAV channel then passes each play data packet to FPGA. FPGA delivers the play data packets to all PEUs.
170 to 184			FPGA Interface logic of PEU stores each broadcast or specifically addressed packet and puts in receive FIFO and interrupts MC. MC reads each play data packet and compares address and stream ID to

	routing table built from stream ID control
messages received from Actual Port Objects.	Payloads of packets with matching stream Ids
play data buffer of DSP for port to which play data directed in response to an interrupt from the DSP that occurs every 26 msec. TDM data for each port taken from TDM bus and moved to TDM data buffer of DSP mapped to port. MC writes any tone data PBX has ordered be	moved to play data buffer of DSP for port to which play data directed in response to an interrupt from the DSP that occurs every 26 msec. TDM data for each port taken from TDM bus and moved to TDM data buffer of DSP mapped to port. MC writes any tone data PBX has ordered be

played on port into tone data buffer of DSP for that port. DSP on each PEU does volume control mixing to weight data from each source in programmable fashion and writes resulting weighted data for each port into outbound TDM data buffer for that port. DSP breaks outbound TDM data into timeslot sized chunks and drives data onto local TDM bus timeslots assigned to port to which data is directed. Data converted to

								analog signals at CODEC of each port.
--	--	--	--	--	--	--	--	---------------------------------------

ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver & FPGA	PEU and DSP
186									User presses DTMF tone that indicate user wants to leave a voice mail.
188									DTMF tones are detected by DSP and DSP interrupts MC. MC generates a "DTMF detected" packet with value then send it to FPGA.
									Pass it to VSR
			Dispatch it to the Actual Port Object mapped to the port at which the DTMF tones were received.						
									Send "DTMF Detected" message to PBX
		PBX invokes TAPI function							

	indicating DTMF detected with value	0	1	2	3	4	5	6	7	8	9	*	#
--	---	---	---	---	---	---	---	---	---	---	---	---	---

ID	IVR APP	PBX	VSR	PEU	WAV Port	Actual	WAV Driver	Device	PEU and
----	---------	-----	-----	-----	----------	--------	------------	--------	---------

(DAL) Object	Port	Driver & FPGA	DSP
			PEU Stop playing data packet by removing Stream ID from the table for addressed port
	Return all buffers waiting to play.		
		Send message to IVR App indicating buffer is done. Repeat for all returned buffers.	
IVR App generates “WODM_UNPREPARE” message with Device ID for all prepared buffers.			
	Either OS or WAV Driver deletes provided buffer space.		

		from the opened file into the buffer. Then App generates “WIDM_ADDBUFFER” message with Device ID and address of buffer. App repeats if buffer is available.	
		Queue buffer.	WAV Driver invokes “Add Buffer” WAV Port Interface function to pass buffer to save recorded data.
198	IVR App generates “WIDM_START” message with Device ID.		
200			WAV Driver invokes “Resume Recording” WAV Port Interface function.
202		WAV Port invokes “Record Start” function of Actual Port Object registered as a listener in this WAV channel.	
204		Actual Port Object generates “Start Sending Record Data” control packet and writes it to	

						shared memory		
ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver & FPGA
208								Device Driver pass it to FPGA and FPGA deliver it to the addressed PEU through Packet Switched Bus or other control channel
210								PEU FPGA BUS interface logic takes control packet off packet bus and passes it to MC.
212								MC reads packet and sets state indicating recording of data from the designated port has started.
214 - 220								DSP interrupts MC every 26 msec. When start recording flag is set, MC responds by retrieving a 208 byte payload section of a record data packet from

			DSP buffer for each port that is recording. MC packetizes data into record data packet by adding header info indicating source and type field indicating it is record data. Packet passed to FPGA which adds CRC bits and transmit over packet switched bus when has a transmit token or is polled for transmission or using timeslot assigned to record data for this port.
222		Device Driver reads FIFO every 26 msec and retrieves record data packet and passes it to VSR dispatcher object	Switch card FPGA receives packet and writes it into FIFO.

ID	IVR APP	PBX	VSR (DAL)	PEU Object	WAV Port	Actual Port	WAV Driver	Device Driver & FPGA	PEU and DSP
224			VSR dispatches message to Actual Port Object that is addressed by source address.						in response to polls asking for new messages
226						Pass address of payload portion of Record Data packet to WAV Port that is registered as listener.			
228					WAV Port copies it to record data buffer.				
230					If buffer is full, return buffer to WAV Driver.				
232						WAV Driver sends message using IVR App callback address specified in the Opening process informing it that Recording Buffer is done.			
234	IVR App invokes system call								

recorded file into area
for recorded voice
mail